

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICATION FOR LETTERS PATENT

**Squared Weil and Tate Pairing Techniques for use with  
Elliptic Curves**

Inventor(s):

**Anne Kirsten Eisentraeger**

**Kristin E. Lauter**

**Peter L. Montgomery**

ATTORNEY'S DOCKET NO. MS1-1276US

## **RELATED PATENT APPLICATIONS**

This patent application is related to co-pending patent application \_\_\_\_ / \_\_\_\_ (Attorney's Docket No. MS1-1275US), titled "Improved Weil and Tate Pairing Techniques Using Parabolas".

## **TECHNICAL FIELD**

This invention relates to cryptography, and more particularly to methods and apparatus that implement improved processing techniques for Weil and Tate pairings using elliptic curves.

## **BACKGROUND**

As computers have become increasingly commonplace in homes and businesses throughout the world, and such computers have become increasingly interconnected via networks (such as the Internet), security and authentication concerns have become increasingly important. One manner in which these concerns have been addressed is the use of a cryptographic technique involving a key-based cipher. Using a key-based cipher, sequences of intelligible data (typically referred to as plaintext) that collectively form a message are mathematically transformed, through an enciphering process, into seemingly unintelligible data (typically referred to as ciphertext). The enciphering can be reversed, allowing recipients of the ciphertext with the appropriate key to transform the ciphertext back to plaintext, while making it very difficult, if not nearly impossible, for those without the appropriate key to recover the plaintext.

Public-key cryptographic techniques are one type of key-based cipher. In public-key cryptography, each communicating party has a public/private key pair.

1 The public key of each pair is made publicly available (or at least available to  
2 others who are intended to send encrypted communications), but the private key is  
3 kept secret. In order to communicate a plaintext message using encryption to a  
4 receiving party, an originating party encrypts the plaintext message into a  
5 ciphertext message using the public key of the receiving party and communicates  
6 the ciphertext message to the receiving party. Upon receipt of the ciphertext  
7 message, the receiving party decrypts the message using its secret private key, and  
8 thereby recovers the original plaintext message.

9 The RSA (Rivest-Shamir-Adleman) method is one well-known example of  
10 public/private key cryptology. To implement RSA, one generates two large prime  
11 numbers  $p$  and  $q$  and multiplies them together to get a large composite number  $N$ ,  
12 which is made public. If the primes are properly chosen and large enough, it will  
13 be practically impossible (i.e., computationally infeasible) for someone who does  
14 not know  $p$  and  $q$  to determine them from knowing only  $N$ . However, in order to  
15 be secure, the size of  $N$  typically needs to be more than 1,000 bits. In some  
16 situations, such a large size makes the numbers too long to be practically useful.

17 One such situation is found in authentication, which can be required  
18 anywhere a party or a machine must prove that it is authorized to access or use a  
19 product or service. An example of such a situation is in a product ID system for a  
20 software program(s), where a user must hand-enter a product ID sequence stamped  
21 on the outside of the properly licensed software package as proof that the software  
22 has been properly paid for. If the product ID sequence is too long, then it will be  
23 cumbersome and user unfriendly.

24 Additionally, not only do software manufacturers lose revenue from  
25 unauthorized copies of their products, but software manufacturers also frequently

1 provide customer support, of one form or another, for their products. In an effort  
2 to limit such support to their licensees, customer support staffs often require a user  
3 to first provide the product ID associated with his or her copy of the product for  
4 which support is sought as a condition for receiving support. Many current  
5 methods of generating product IDs, however, have been easily discerned by  
6 unauthorized users, allowing product IDs to be generated by unauthorized users.

7         Given the apparent ease with which unauthorized users can obtain valid  
8 indicia, software manufacturers are experiencing considerable difficulty in  
9 discriminating between licensees and such unauthorized users in order to provide  
10 support to the former while denying it to the latter. As a result, manufacturers  
11 often unwittingly provide support to unauthorized users, thus incurring additional  
12 and unnecessary support costs. If the number of unauthorized users of a software  
13 product is sufficiently large, then these excess costs associated with that product  
14 can be quite significant.

15         New curve-based cryptography techniques have recently been employed to  
16 allow software manufacturers to appreciably reduce the incidence of unauthorized  
17 copying of software products. For example, product IDs have been generated  
18 using elliptic curve cryptographic techniques. The resulting product IDs provide  
19 improved security. Curve-based cryptographic techniques may also be used to  
20 perform other types of cryptographic services.

21         As curve-based cryptosystems grow in popularity, it would be useful to  
22 have new and improved techniques for performing the computations associated  
23 with the requisite mathematical operations. Hence, there is a continuing need for  
24 improved mathematical and/or computational methods and apparatus in curve-based  
25 cryptosystems.

## SUMMARY

Methods and apparatus are provided for determining “Squared Weil pairings” and/or “Squared Tate pairings” based on an elliptic curve, for example, and which are then used to support cryptographic processing of selected information. Significant improvements are provided in computing efficiency over conventional implementations of the Weil and Tate pairings. The resulting Squared Weil and/or Tate pairings can be substituted for conventional Weil or Tate pairings in a variety of applications.

The above stated needs and/or others are met, for example, by a method that includes selecting an elliptic curve, determining a Squared Weil pairing based on the elliptic curve, and cryptographically processing selected information based on the Squared Weil pairing.

Here, for example, an elliptic curve  $E$  over a field  $K$  may be used, wherein  $E$  can be represented as an equation  $y^2 = x^3 + ax + b$ . The method may then include establishing a point **id** that is defined as a point at infinity on  $E$ , and wherein **P**, **Q**, **R**, **X** are points on  $E$  wherein **X** is an indeterminate denoting an independent variable of a function, and wherein  $x(\mathbf{X})$ ,  $y(\mathbf{X})$  are functions mapping the point **X** on  $E$  to its affine  $x$  and  $y$  coordinates, and wherein a line passes through any three points **P**, **Q**, **R** which satisfy **P + Q + R = id**. Note, as used herein, a bold **+** or **−** operator denotes arithmetic in the elliptic curve group, whereas a normal  $+$  or  $-$  operator denotes arithmetic in the field  $K$  or in the integers.

In certain implementations,  $\mathbf{P}$  and  $\mathbf{Q}$  are  $m$ -torsion points on  $E$  and  $m$  is an odd prime, and the method further includes determining the squared Weil pairing based on

$$\frac{f_{m,\mathbf{P}}(\mathbf{Q})f_{m,\mathbf{Q}}(-\mathbf{P})}{f_{m,\mathbf{P}}(-\mathbf{Q})f_{m,\mathbf{Q}}(\mathbf{P})} = -e_m(\mathbf{P}, \mathbf{Q})^2,$$

where  $e_m$  denotes the Weil-pairing.

Another exemplary method includes determining a Squared Weil Pairing  $e_m(\mathbf{P}, \mathbf{Q})^2$  by establishing an odd prime  $m$  and a curve  $E$ ; and based on two  $m$ -torsion points  $\mathbf{P}$  and  $\mathbf{Q}$  on  $E$ , computing  $e_m(\mathbf{P}, \mathbf{Q})^2$ . In certain further implementations, the method also includes forming a mathematical chain for  $m$ . For example, an addition chain or an addition-subtraction chain may be formed such that every element in the mathematical chain is a sum or difference of two earlier elements in the mathematical chain, which continues until  $m$  is included.

Then, for example, for each  $j$  in the mathematical chain, a tuple

$t_j = [j\mathbf{P}, j\mathbf{Q}, n_j, d_j]$  is formed such that

$$\frac{n_j}{d_j} = \frac{f_{j,\mathbf{P}}(\mathbf{Q})f_{j,\mathbf{Q}}(-\mathbf{P})}{f_{j,\mathbf{P}}(-\mathbf{Q})f_{j,\mathbf{Q}}(\mathbf{P})}.$$

In accordance with still other exemplary implementations, another method includes determining a Squared Weil pairing  $(m, \mathbf{P}, \mathbf{Q})$ , where  $m$  is an odd prime number, by setting  $t_1 = [\mathbf{P}, \mathbf{Q}, 1, 1]$ , using an addition-subtraction chain to determine  $t_m = [m\mathbf{P}, m\mathbf{Q}, n_m, d_m]$ , and if  $n_m$  and  $d_m$  are nonzero, then determining:

$$\frac{n_m}{d_m} = \frac{f_{m,\mathbf{P}}(\mathbf{Q})f_{m,\mathbf{Q}}(-\mathbf{P})}{f_{m,\mathbf{P}}(-\mathbf{Q})f_{m,\mathbf{Q}}(\mathbf{P})},$$

and cryptographically processing selected information based on the Squared Weil pairing.

Another exemplary method includes selecting an elliptic curve, determining Squared Tate pairing based on the elliptic curve, and cryptographically processing selected information based on the Squared Tate pairing. Here, for example, an elliptic curve  $E$  over a field  $K$  with  $q = p^n$  elements may be used, wherein  $E$  can be represented as an equation  $y^2 = x^3 + ax + b$ . Letting  $m$  be an odd prime dividing  $q-1$  and  $\mathbf{P}$  be an  $m$ -torsion point on  $E$ . Let  $\mathbf{Q} \neq \mathbf{id}$  be a point on  $E$ . If  $\mathbf{P}$  is not equal to a multiple of  $\mathbf{Q}$ , the method may further include determining

$$\left( \frac{f_{m,\mathbf{P}}(\mathbf{Q})}{f_{m,\mathbf{P}}(-\mathbf{Q})} \right)^{\frac{q-1}{m}} = v_m(\mathbf{P}, \mathbf{Q}),$$

where  $v_m$  denotes the squared Tate-pairing.

## **BRIEF DESCRIPTION OF THE DRAWINGS**

The present invention is illustrated by way of example and not limitation in the figures of the accompanying drawings. The same numbers are used throughout the figures to reference like components and/or features.

1 Fig. 1 is a block diagram illustrating an exemplary cryptosystem in  
2 accordance with certain implementations of the present invention.

3 Fig. 2 illustrates an exemplary system using a product identifier to validate  
4 software in accordance with certain implementations of the present invention.

5 Fig. 3 illustrates an exemplary process for use in curve-based  
6 cryptosystems in accordance with certain implementations of the present  
7 invention.

8 Fig. 4 illustrates a more general exemplary computer environment which  
9 can be used in various implementations of the invention.

## 10 11 **DETAILED DESCRIPTION**

### 12 **Introduction**

13 The discussions herein assume a basic understanding of cryptography by  
14 the reader. For a basic introduction to cryptography, the reader is directed to a  
15 book written by Bruce Schneier and entitled "Applied Cryptography: Protocols,  
16 Algorithms, and Source Code in C," published by John Wiley & Sons with  
17 copyright 1994 (or second edition with copyright 1996).

18 Described herein are techniques that can be used with a curve-based  
19 cryptosystem, and in particular elliptic curve-based cryptosystems. In certain  
20 examples, the techniques take the form of methods and apparatus that can be  
21 implemented in logic within one or more devices. One such device, for example,  
22 is a computing device that is configured to perform at least a portion of the  
23 processing required for a particular cryptographic capability or application.

24 The techniques provided herein can be implemented and/or otherwise  
25 adapted for use in a variety of cryptographic capabilities and applications. By way



1 of example, the techniques may be employed to support: key generation logic,  
2 e.g., for one-round three-way key establishment applications; identity-based  
3 encryption logic; short signature logic, e.g., product identifier logic; and/or other  
4 like cryptographic logic.

5 The term logic as used herein is meant to include any suitable form of logic  
6 that may be employed. Thus, for example, logic may include hardware, firmware,  
7 software, or any combination thereof.

8 The term curve-based cryptosystem as used herein refers to logic that at  
9 least partially provides for curve-based encryption and/or decryption using key(s)  
10 that are generated based at least partially on aspects or characteristics of an elliptic  
11 curve or other like curve.

12 Such curve-based cryptosystems can be used to encrypt any of a wide  
13 variety of information. Here, for example, one exemplary cryptosystem is  
14 described primarily with respect to generation of a short signature or product  
15 identifier, which is a code that allows validation and/or authentication of a  
16 machine, program, user, etc. The signature is a "short" signature in that it uses a  
17 relatively small number of characters.

18 With this in mind, attention is drawn to Fig. 1, which is a block diagram  
19 illustrating an exemplary cryptosystem 100 in accordance with certain  
20 implementations of the present invention. Cryptosystem 100 includes an  
21 encryptor 102 and a decryptor 104. A plaintext message 106 is received at an  
22 input module 108 of encryptor 102, which is a curve-based encryptor that encrypts  
23 message 106 based on a public key generated based on a secret known by  
24 decryptor 104. Plaintext message 106 is typically an unencrypted message,  
25 although encryptor 102 can encrypt any type of message/data. Thus, message 106

1 may alternatively be encrypted or encoded by some other component (not shown)  
2 or a user.

3 An output module 110 of encryptor 102 outputs the encrypted version of  
4 plaintext message 106, which is ciphertext 112. Ciphertext 112 can then be  
5 communicated to decryptor 104, which can be implemented, for example, on a  
6 computer system remote from a computer system on which encryptor 102 is  
7 implemented. Given the encrypted nature of ciphertext 112, the communication  
8 link between encryptor 102 and 104 need not be secure (it is typically presumed  
9 that the communication link is not secure). The communication link can be any of  
10 a wide variety of public and/or private networks implemented using any of a wide  
11 variety of conventional public and/or proprietary protocols, and including both  
12 wired and wireless implementations. Additionally, the communication link may  
13 include other non-computer network components, such as hand-delivery of media  
14 including ciphertext or other components of a product distribution chain.

15 Decryptor 104 receives ciphertext 112 at input module 114 and, being  
16 aware of the secret used to encrypt message 106, is able to readily decrypt  
17 ciphertext 112 to recover the original plaintext message 106, which is output by  
18 output module 116 as plaintext message 118. Decryptor 104 is a curve-based  
19 decryptor that decrypts the message based on the same curve as was used by  
20 encryptor 102.

21 Encryption and decryption are performed in cryptosystem 100 based on a  
22 secret, such as points on the elliptic curve. This secret is known to decryptor 104,  
23 and a public key generated based on the secret is known to encryptor 102. This  
24 knowledge allows encryptor 102 to encrypt a plaintext message that can be  
25 decrypted only by decryptor 104. Other components, including encryptor 102,

1 which do not have knowledge of the secret cannot decrypt the ciphertext (although  
2 decryption may be technically possible, it is not computationally feasible).  
3 Similarly, decryptor 104 can also generate a message using the secret and based on  
4 a plaintext message, a process referred to as digitally signing the plaintext  
5 message. This signed message can then be communicated to other components,  
6 such as encryptor 102, which can in turn verify the digital signature based on the  
7 public key.

8 Fig. 2 illustrates an exemplary system using a product identifier to validate  
9 software in accordance with certain implementations of the present invention. Fig.  
10 2 illustrates a software copy generator 120 including a product identifier (ID)  
11 generator 122. Software copy generator 120 produces software media 124 (e.g., a  
12 CD-ROM, DVD (Digital Versatile Disk)) that contains typically all the files  
13 needed to collectively implement a complete copy of one or more application  
14 programs, (e.g., a word processing program, a spreadsheet program, an operating  
15 system, a suite of programs). These files are received from source files 126,  
16 which may be a local source (e.g., a hard drive internal to generator 120), a remote  
17 source (e.g., coupled to generator 120 via a network), or a combination thereof.  
18 Although only a single generator 120 is illustrated in Fig. 2, typically multiple  
19 such generators operate individually and/or cooperatively to increase the rate at  
20 which software media 124 can be generated.

21 Product ID generator 122 generates a product ID 128 that can include  
22 numbers, letters, and/or other symbols. Generator 122 generates product ID 128  
23 using the curve-based encryption techniques described herein. The product ID  
24 128 is typically printed on a label and affixed to either a carrier containing  
25 software media 124 or a box into which software media 124 is placed.

1 Alternatively, the product ID 128 may be made available electronically, such as a  
2 certificate provided to a user when receiving a softcopy of the application program  
3 via an on-line source (e.g., downloading of the software via the Internet). The  
4 product ID can serve multiple functions. First, the product ID can be  
5 cryptographically validated in order to verify that the product ID is a valid product  
6 ID (and thus allowing, for example, the application program to be installed).  
7 Additionally, the product ID can optionally serve to authenticate the particular  
8 software media 124 to which it is associated.

9 The generated software media 124 and associated product ID 128 are then  
10 provided to a distribution chain 130. Distribution chain 130 represents any of a  
11 variety of conventional distribution systems and methods, including possibly one  
12 or more "middlemen" (e.g., wholesalers, suppliers, distributors, retail stores (either  
13 on-line or brick and mortar), etc.). Regardless of the manner in which media 124  
14 and the associated product ID 128 are distributed, eventually media 124 and  
15 product ID 128 are purchased (e.g., licensed), by the user of a client computer 132.

16 Client computer 132 includes a media reader 134 capable of reading  
17 software media 124 and installing the application program onto client computer  
18 132 (e.g., installing the application program on to a hard disk drive (not shown) of  
19 client computer 132). Part of this installation process involves entry of the product  
20 ID 128. This entry may be a manual entry (e.g., the user typing in the product ID  
21 via a keyboard), or alternatively an automatic entry (e.g., computer 132  
22 automatically accessing a particular field of a license associated with the  
23 application program and extracting the product ID there from). Client computer  
24 132 also includes a product ID validator 136 which validates, during installation of  
25

1 the application program, the product ID 128. This validation is performed using  
2 the curve-based decryption techniques.

3 If validator 136 determines that the product ID is valid, then an appropriate  
4 course of action is taken (e.g., an installation program on software media 124  
5 allows the application to be installed on computer 132). However, if validator 136  
6 determines that the product ID is invalid, then a different course of action is taken  
7 (e.g., the installation program terminates the installation process preventing the  
8 application program from being installed).

9 Product ID validator 136 also optionally authenticates the application  
10 program based on the product ID 128. This authentication verifies that the product  
11 ID 128 entered at computer 132 corresponds to the particular copy of the  
12 application being accessed. The authentication can be performed at different  
13 times, such as during installation, or when requesting product support or an  
14 upgrade. Alternatively, this authentication may be performed at a remote location  
15 (e.g., at a call center when the user of client computer 132 calls for technical  
16 support, the user may be required to provide the product ID 128 before receiving  
17 assistance).

18 If the application program manufacturer desires to utilize the authentication  
19 capabilities of the product ID, then the product ID generated by generator 122 for  
20 each copy of an application program is unique. This uniqueness is created by  
21 assigning a different initial number or value to each copy of the application  
22 program. This initial value can then be used as a basis for generating the product  
23 ID.

24 The unique value associated with the copy of the application program can  
25 be optionally maintained by the manufacturer as an authentication record 138

1 (e.g., a database or list) along with an indication of the particular copy of the  
2 application program. This indication can be, for example, a serial number  
3 embedded in the application program or on software media 124, and may be  
4 hidden in any of a wide variety of conventional manners.

5 Alternatively, the individual number itself may be a serial number that is  
6 associated with the particular copy, thereby allowing the manufacturer to verify  
7 the authenticity of an application program by extracting the initial value from the  
8 product ID and verifying that it is the same as the serial number embedded in the  
9 application program or software media 124.

10 Appropriate action can be taken based on whether the product ID is  
11 authenticated. These actions can vary, depending on the manufacturer's desires  
12 and/or action being taken at computer 132 that caused the authentication check to  
13 occur. For example, if a user is attempting to install an application program then  
14 installation of the program may be allowed only if the authentication succeeds. By  
15 way of another example, the manufacturer's support technicians may provide  
16 assistance to a user of computer 132 only if the authentication succeeds, or an  
17 upgrade version of the application program may be installed only if authentication  
18 of the previous version of the application program succeeds.

19 The logic of certain curve-based cryptosystems utilizes what are commonly  
20 referred to as "Weil and Tate pairings" during a signature verification process  
21 when using elliptic curves. The Weil and Tate pairings have been proposed for use  
22 in many aspects of cryptography. They may be used, for example, to form  
23 efficient protocols to do one-round three-way key establishment, identity-based  
24 encryption, short signatures, and the like.

1        It is important, however, given the amount of processing to have efficient  
2 implementations of the Weil and Tate pairings to cut down on the cost of  
3 implementing these protocols. Computation of the Weil or Tate pairing in  
4 conventional cryptosystems typically follows “Miller’s algorithm”, which is  
5 described, for example, in “Identity-Based Encryption From The Weil Pairing”, by  
6 Dan Boneh and Matthew Franklin, published in SIAM J. of Computing, Vol. 32,  
7 No. 3, pp. 586-615, 2003.

8        As is well-known, for a fixed positive integer  $m$ , the Weil pairing  $e_m$  is a  
9 bilinear map that takes as input two  $m$ -torsion points on an elliptic curve, and  
10 outputs an  $m^{th}$  root of unity. For elliptic curves, as is well-known, the Tate pairing  
11 is related to the Weil pairing by the fact that the Weil pairing is a quotient of the  
12 output of two applications of the Tate pairing. The algorithms for these pairings  
13 depend on constructing rational functions with a prescribed pattern of poles and  
14 zeros.

15        The Miller algorithm as typically implemented in conventional curve-based  
16 cryptosystems calls for the evaluation of the Weil or Tate pairing by evaluating a  
17 function at two selected points on the elliptic curve, wherein one of the points is  
18 formed from a “random” point selected using a randomly generated input.  
19 Unfortunately, there is a chance that the Miller algorithm will essentially fail with  
20 some random input values, due to a division by zero. When the Miller algorithm  
21 fails, then the logic will usually need to rerun the Miller algorithm using a  
22 different random input value. Although the failure rate of the Miller algorithm  
23 tends to be fairly low, if it is required to be run thousands or millions of times,  
24 eventually the processing delays may become significant. Also, if the  
25 computations are performed in a parallel processing environment, the completions

1 may be slowed or delayed as some of the processing pipelines or the like are  
2 required to rerun the Miller algorithm. Real-time applications, which have rigid  
3 bounds on maximum computation time rather than average computation time, may  
4 not be able to afford the reruns.

5 The improved techniques described herein provide increased efficiency and  
6 an alternative method to the standard methods which have been proposed. For  
7 example, in accordance with certain aspects of the present invention, the improved  
8 techniques do not require a randomly chosen  $m$ -torsion point as described above  
9 and as such always generate a correct answer.

10 With this exemplary improvement in mind, in the following sections an  
11 improved algorithm is described for computing what is hereby referred to as the  
12 “squared Weil pairing”,  $e_m(\mathbf{P}, \mathbf{Q})^2$ . With the squared Weil pairing, one may obtain  
13 a significant speed-up over the standard implementation of the Weil pairing  
14 (Miller’s algorithm). While the Tate pairing is already more efficient to  
15 implement than the Weil pairing, the improved techniques herein can also be  
16 employed in an improved algorithm for computing the “squared Tate pairing” for  
17 elliptic curves. The squared Weil pairing and/or the squared Tate pairing can be  
18 substituted for the Weil or Tate pairings in any of the above applications.

19 By way of further reference, other exemplary curve-based cryptosystems  
20 are provided in the following references: “Short Signatures from the Weil  
21 Pairing”, by Dan Boneh, et al., in *Advances in Cryptography – Asiacrypt 2001*,  
22 Lecture Notes in Computer Science, Vol. 2248, Springer-Verlag, pp. 514-532;  
23 and, “The Weil and Tate Pairings as Building Blocks for Public Key  
24 Cryptosystems (Survey)”, by Antoine Joux, in *Algorithmic Number Theory, 5<sup>th</sup>*  
25 *International Symposium ANTS-V, Sydney, Australia, July 2002 proceedings*,



Claus Fieker and David R. Kohel (Eds.), Lecture Notes in Computer Science, Vol. 2369, Springer-Verlag, pp. 20-32.

Attention is now drawn to Fig. 3, which is a flow diagram illustrating an exemplary process 150 for use in comparing the Weil and Tate pairings for elliptic curves. In act 152, an addition chain, addition-subtraction chain, or the like, is formed for  $m$ , wherein  $m$  is an integer greater than zero and an  $m$ -torsion point  $\mathbf{P}$  is fixed on an elliptic curve  $E$ . In act 154,  $((j+k)\mathbf{P}, f_{j+k, \mathbf{P}}(\mathbf{X}))$  is determined using  $(j\mathbf{P}, f_{j, \mathbf{P}}(\mathbf{X}))$  and  $(k\mathbf{P}, f_{k, \mathbf{P}}(\mathbf{X}))$ , wherein  $j$  and  $k$  are integers,  $j\mathbf{P}$ ,  $k\mathbf{P}$  and  $(j+k)\mathbf{P}$  are multiples of point  $\mathbf{P}$  and  $f_{j, \mathbf{P}}(\mathbf{X})$ ,  $f_{k, \mathbf{P}}(\mathbf{X})$  and  $f_{j+k, \mathbf{P}}(\mathbf{X})$  are functions in the indeterminate  $\mathbf{X}$ , and  $((j+k)\mathbf{P}, f_{j+k, \mathbf{P}}(\mathbf{X}))$  represents an iterative building block for forming the output of the pairing via a chain. With Weil pairing, for example,  $((j+k)\mathbf{P}, f_{j+k, \mathbf{P}}(\mathbf{X}))$  can also be run with  $\mathbf{P}$  replaced by another  $m$ -torsion point  $\mathbf{Q}$ , i.e.,  $((j+k)\mathbf{Q}, f_{j+k, \mathbf{Q}}(\mathbf{X}))$ . In act 156,  $h_{j+k}$  is determined given  $h_j$  and  $h_k$ , wherein  $h_j$ ,  $h_k$  and  $h_{j+k}$  are field elements and for example,

$$h_j = f_{j, \mathbf{P}}(\mathbf{Q}_1) / f_{j, \mathbf{P}}(\mathbf{Q}_2)$$

for certain points  $\mathbf{Q}_1$  and  $\mathbf{Q}_2$  (independent of  $j$ ) on  $E$  and the goal is to compute  $h_m$ . In conventional Miller algorithms  $\mathbf{Q}_1$  and  $\mathbf{Q}_2$  are random value inputs.

In certain improvements provided herein, for example, an improved algorithm essentially produces:

$$h'_j := f_{j, \mathbf{P}}(\mathbf{Q}) / f_{j, \mathbf{P}}(-\mathbf{Q})$$

wherein  $-\mathbf{Q} = \mathbf{id} - \mathbf{Q}$  is the complement of  $\mathbf{Q}$ . When the elliptic curve is given as  $E : y^2 = x^3 + ax + b$ , then the complement of a point is the reflection of the point over the  $x$ -axis, which for an elliptic curve is the additive inverse of the point.

1 In accordance with certain aspects of the present invention, an improvement  
2 is made to act 156 wherein squared Weil pairing is introduced. In accordance with  
3 certain further aspects of the present invention, an improvement is made to a  
4 subprocess of act 154 wherein a parabola is introduced for computing Weil and  
5 Tate pairings in a manner that reduces the number of computation steps required.  
6 Improved cryptosystems may implement either one or both of these aspects of the  
7 present invention.

### 8 9 Squared Weil Pairing for Elliptic Curves

10 The purpose of this section is to construct a new pairing, referred to as  
11 squared Weil pairing, which has the advantage of being more efficient to compute  
12 than Miller's algorithm for the original Weil pairing. The improved algorithm  
13 presented herein has the advantage that it is guaranteed to output the correct  
14 answer since it does not depend on inputting a randomly chosen  $m$ -torsion point.  
15 As mentioned earlier, certain conventional implementations of Miller's algorithm  
16 sometimes require more than one iteration of the algorithm, since the randomly  
17 chosen  $m$ -torsion point may cause the algorithm to fail at times.

18 Let  $E : y^2 = x^3 + ax + b$  be an elliptic curve over a field  $K$ , not of  
19 characteristic 2 or 3. Characteristic 2 and 3 can be handled by introducing the  
20 more general equation for an elliptic curve.

21 Introducing some further notation, let:

22  **$\text{id}$**  be the point at infinity on  $E$ ;

23  **$P, Q, R, X$**  (capitals) be points on  $E$ , wherein  **$X$**  is an indeterminate  
24 denoting the (main) independent variable of a function;  
25

1  $x(\mathbf{X}), y(\mathbf{X})$  be (rational) functions mapping a point  $\mathbf{X}$  on  $E$  to its (affine)  $x$   
2 and  $y$ -coordinates;

3 Let line  $(\mathbf{P}, \mathbf{Q}, \mathbf{R})(\mathbf{X})$  denote the equation of the line (linear in  $x(\mathbf{X})$  and  
4  $y(\mathbf{X})$ ) passing through three points  $\mathbf{P}, \mathbf{Q}, \mathbf{R}$  on  $E$ , which satisfy  $\mathbf{P} + \mathbf{Q} + \mathbf{R} = \mathbf{id}$ ,  
5 and wherein when two of  $\mathbf{P}, \mathbf{Q}, \mathbf{R}$  are equal, this is a tangent line at that common  
6 point.

### 8 Function $f_{j, \mathbf{P}}$ and its Construction

9 If  $j$  is an integer and  $\mathbf{P}$  a point on  $E$ , then  $f_{j, \mathbf{P}}$  and  $f_{j, \mathbf{P}}(\mathbf{X})$  will refer to a  
10 rational function on  $E$  whose divisor of zeros and poles is:

$$11 \quad (f_{j, \mathbf{P}}) = j(\mathbf{P}) - (j\mathbf{P}) - (j-1)(\mathbf{id}),$$

12 where parentheses around a point on  $E$  indicate that it is being considered formally  
13 as a point on  $E$ . If  $j > 1$  and  $\mathbf{P}, j\mathbf{P}$ , and  $\mathbf{id}$  are distinct, then  $f_{j, \mathbf{P}}(\mathbf{X})$  has a  $j$ -fold zero  
14 at  $\mathbf{X} = \mathbf{P}$ , a simple pole at  $\mathbf{X} = j\mathbf{P}$ , a  $(j-1)$ -fold pole at infinity (i.e., at  $\mathbf{X} = \mathbf{id}$ ), and  
15 no other poles or zeros.

16 The theory of divisors states that  $f_{j, \mathbf{P}}$  exists and is unique up to a nonzero  
17 scale factor (multiplicative constant). If  $\mathbf{Q}_1$  and  $\mathbf{Q}_2$  are given, then the quotient  
18  $f_{j, \mathbf{P}}(\mathbf{Q}_1) / f_{j, \mathbf{P}}(\mathbf{Q}_2)$  is well-defined unless a division by zero occurs.

19 When  $j = 0$  or  $j = 1$ ,  $f_{j, \mathbf{P}}$  can be any nonzero constant.

20 If one knows  $f_{j, \mathbf{P}}$  and  $f_{k, \mathbf{P}}$  for two integers  $j$  and  $k$ , then a simple, well-  
21 known, construction gives  $f_{-j-k, \mathbf{P}}$ . One wants  $f_{-j-k, \mathbf{P}}$  to satisfy

$$22 \quad (f_{-j-k, \mathbf{P}} f_{j, \mathbf{P}} f_{k, \mathbf{P}}) = (f_{-j-k, \mathbf{P}}) + (f_{j, \mathbf{P}}) + (f_{k, \mathbf{P}}) = 3(\mathbf{id}) - ((-j-k)\mathbf{P}) - (j\mathbf{P}) - (k\mathbf{P}).$$

23 This will be satisfied if we choose  $f_{-j-k, \mathbf{P}}$  so that:

$$24 \quad f_{-j-k, \mathbf{P}}(\mathbf{X}) f_{j, \mathbf{P}}(\mathbf{X}) f_{k, \mathbf{P}}(\mathbf{X}) \text{line}(j\mathbf{P}, k\mathbf{P}, (-j-k)\mathbf{P})(\mathbf{X}) = \text{constant}.$$

Then repeating this construction on  $f_{0, \mathbf{P}}$  and  $f_{-j-k, \mathbf{P}}$  gives  $f_{j+k, \mathbf{P}}$ . The line through  $0\mathbf{P} = \mathbf{id}$ ,  $(-j-k)\mathbf{P}$ , and  $(j+k)\mathbf{P}$  is vertical (i.e., its equation does not reference the  $y$ -coordinate). This results in the useful constructions

$$f_{j+k, \mathbf{P}}(\mathbf{X}) = f_{j, \mathbf{P}}(\mathbf{X}) f_{k, \mathbf{P}}(\mathbf{X}) \frac{\text{line}(j\mathbf{P}, k\mathbf{P}, (-j-k)\mathbf{P})(\mathbf{X})}{\text{line}(\mathbf{id}, (-j-k)\mathbf{P}, (j+k)\mathbf{P})(\mathbf{X})}$$

$$f_{j-k, \mathbf{P}}(\mathbf{X}) = \frac{f_{j, \mathbf{P}}(\mathbf{X}) \text{line}(\mathbf{id}, j\mathbf{P}, -j\mathbf{P})(\mathbf{X})}{f_{k, \mathbf{P}}(\mathbf{X}) \text{line}(-j\mathbf{P}, k\mathbf{P}, (j-k)\mathbf{P})(\mathbf{X})}$$

Other possibly useful formulae include:

$$f_{j, \mathbf{id}} = \text{constant};$$

$$f_{j, -\mathbf{P}}(\mathbf{X}) = f_{j, \mathbf{P}}(-\mathbf{X}) * (\text{constant});$$

If  $(\mathbf{P} + \mathbf{Q} + \mathbf{R} = \mathbf{id})$ , then:

$$f_{j, \mathbf{P}}(\mathbf{X}) f_{j, \mathbf{Q}}(\mathbf{X}) f_{j, \mathbf{R}}(\mathbf{X}) = \frac{\text{line}(\mathbf{P}, \mathbf{Q}, \mathbf{R})(\mathbf{X})^j}{\text{line}(j\mathbf{P}, j\mathbf{Q}, j\mathbf{R})(\mathbf{X})}.$$

### Squared Weil-Pairing Formula

Let  $m$  be an odd prime. Suppose  $\mathbf{P}$  and  $\mathbf{Q}$  are  $m$ -torsion points on  $E$ , with neither being the identity and  $\mathbf{P}$  not equal to  $\pm\mathbf{Q}$ .

Then

$$\frac{f_{m, \mathbf{P}}(\mathbf{Q}) f_{m, \mathbf{Q}}(-\mathbf{P})}{f_{m, \mathbf{P}}(-\mathbf{Q}) f_{m, \mathbf{Q}}(\mathbf{P})} = -e_m(\mathbf{P}, \mathbf{Q})^2$$

where  $e_m$  denotes the Weil-pairing.

### Exemplary Algorithm for $e_m(\mathbf{P}, \mathbf{Q})^2$

Fix an odd prime  $m$  and the curve  $E$ . Given two  $m$ -torsion points  $\mathbf{P}$  and  $\mathbf{Q}$  on  $E$ , one needs to compute  $e_m(\mathbf{P}, \mathbf{Q})^2$ .

In accordance with certain exemplary implementations of the present invention, the algorithm includes forming an addition or addition-subtraction chain for  $m$ . That is, after an initial 1, every element in the chain is a sum or difference of two earlier elements in the chain, until an  $m$  appears. Well-known techniques, give a chain of length  $O(\log(m))$ .

For each  $j$  in the addition-subtraction chain, form a tuple

$$t_j = [j\mathbf{P}, j\mathbf{Q}, n_j, d_j]$$

such that  $n_j$  and  $d_j$  are field elements such that

$$\frac{n_j}{d_j} = \frac{f_{j,\mathbf{P}}(\mathbf{Q})f_{j,\mathbf{Q}}(-\mathbf{P})}{f_{j,\mathbf{P}}(-\mathbf{Q})f_{j,\mathbf{Q}}(\mathbf{P})}.$$

Keeping track of the numerators and denominators separately until the end is optional.

To do this, start with  $t_1 = [\mathbf{P}, \mathbf{Q}, 1, 1]$ . Given  $t_j$  and  $t_k$ , this procedure gets  $t_{j+k}$ :

form elliptic curve sums:  $j\mathbf{P} + k\mathbf{P} = (j+k)\mathbf{P}$  and  $j\mathbf{Q} + k\mathbf{Q} = (j+k)\mathbf{Q}$ ;

find line:  $\text{line}(j\mathbf{P}, k\mathbf{P}, (-j-k)\mathbf{P})(\mathbf{X}) = c_0 + c_1x(\mathbf{X}) + c_2y(\mathbf{X})$ ;

find line:  $\text{line}(j\mathbf{Q}, k\mathbf{Q}, (-j-k)\mathbf{Q})(\mathbf{X}) = c_0' + c_1'x(\mathbf{X}) + c_2'y(\mathbf{X})$ .

Set:

$$n_{j+k} = n_j * n_k * (c_0 + c_1x(\mathbf{Q}) + c_2y(\mathbf{Q})) * (c_0' + c_1'x(\mathbf{P}) - c_2'y(\mathbf{P}))$$

and

$$d_{j+k} = d_j * d_k * (c_0 + c_1x(\mathbf{Q}) - c_2y(\mathbf{Q})) * (c_0' + c_1'x(\mathbf{P}) + c_2'y(\mathbf{P})).$$

A similar construction gives  $t_{j-k}$  from  $t_j$  and  $t_k$ . Observe that the vertical lines through  $(j+k)\mathbf{P}$  and  $(j+k)\mathbf{Q}$  do not appear in the formulae for  $n_{j+k}$  and  $d_{j+k}$ , —

1 this is because the contributions from  $\mathbf{Q}$  and  $-\mathbf{Q}$  (or from  $\mathbf{P}$  and  $-\mathbf{P}$ ) are equal.

2 Here  $-\mathbf{Q}$  is the complement of  $\mathbf{Q}$  and  $-\mathbf{P}$  is the complement of  $\mathbf{P}$ .

3 When  $j + k = m$ , one can further simplify this to  $n_{j+k} = n_j * n_k$  and  $d_{j+k} = d_j *$   
 4  $d_k$ , since  $c_2$  and  $c_2'$  will be zero.

5 Pseudocode may take the following form, for example:

```

6 procedure Squared_Weil_Pairing( $m, \mathbf{P}, \mathbf{Q}$ )
7   issue an error if  $m$  is not an odd prime.
8   if ( $\mathbf{P} = \mathbf{id}$  or  $\mathbf{Q} = \mathbf{id}$  or  $\mathbf{P} = \pm \mathbf{Q}$ ) then
9     return 1;
10  else
11     $t_1 = [\mathbf{P}, \mathbf{Q}, 1, 1]$ ;
12    use an addition-subtraction chain to get
13     $t_m = [m\mathbf{P}, m\mathbf{Q}, n_m, d_m]$ .
14    issue an error if  $m\mathbf{P}$  or  $m\mathbf{Q}$  is not  $\mathbf{id}$ .
15    if ( $n_m = 0$  or  $d_m = 0$ ) then
16      return 1;
17    else
18      return  $-n_m/d_m$ ;
19    end if;
20  end if;

```

16 When  $n_m$  and  $d_m$  are nonzero, then the computation

$$\frac{n_m}{d_m} = \frac{f_{m,\mathbf{P}}(\mathbf{Q})f_{m,\mathbf{Q}}(-\mathbf{P})}{f_{m,\mathbf{P}}(-\mathbf{Q})f_{m,\mathbf{Q}}(\mathbf{P})}$$

19 has been successful, and the output is correct. If, however, some  $n_m$  or  $d_m$  is zero,  
 20 then some factor such as  $c_0 + c_1 * x(\mathbf{Q}) + c_2 * y(\mathbf{Q})$  must have vanished. That line  
 21 was chosen to pass through  $j\mathbf{P}$ ,  $k\mathbf{P}$ , and  $(-j-k)\mathbf{P}$ , for some  $j$  and  $k$ .

22 This factor does not vanish at any other point on the elliptic curve.  
 23 Therefore this factor can vanish only if  $\mathbf{Q} = j\mathbf{P}$  or  $\mathbf{Q} = k\mathbf{P}$  or  $\mathbf{Q} = (-j-k)\mathbf{P}$  for  
 24 some  $j$  and  $k$ . In all of these cases  $\mathbf{Q}$  will be a multiple of  $\mathbf{P}$ , ensuring that  
 25

$$e_m(\mathbf{P}, \mathbf{Q}) = 1.$$

### Squared Tate Pairing For Elliptic Curves

#### Squared Tate Pairing Formula

Let  $m$  be an odd prime. Suppose  $\mathbf{P}$  is an  $m$ -torsion point on  $E$ , and  $\mathbf{Q}$  is a point on the curve, with neither being the identity and  $\mathbf{P}$  not equal to a multiple of  $\mathbf{Q}$ . Assume that  $E$  is defined over  $K$ , where  $K$  has  $q = p^n$  elements and suppose  $m$  divides  $q-1$ . Then

$$\left( \frac{f_{m,\mathbf{P}}(\mathbf{Q})}{f_{m,\mathbf{P}}(-\mathbf{Q})} \right)^{\frac{q-1}{m}} = v_m(\mathbf{P}, \mathbf{Q})$$

where  $v_m$  denotes the squared Tate-pairing.

#### Exemplary Algorithm For $v_m(\mathbf{P}, \mathbf{Q})$

Fix an odd prime  $m$  and the curve  $E$ . Given an  $m$ -torsion point  $\mathbf{P}$  on  $E$  and a point  $\mathbf{Q}$  on  $E$ , one needs to compute  $v_m(\mathbf{P}, \mathbf{Q})$ .

As before, one starts with an addition or addition-subtraction chain for  $m$ .

For each  $j$  in the addition-subtraction chain, one then forms a tuple

$$t_j = [j\mathbf{P}, n_j, d_j]$$

such that

$$\frac{n_j}{d_j} = \frac{f_{j,\mathbf{P}}(\mathbf{Q})}{f_{j,\mathbf{P}}(-\mathbf{Q})}.$$

Keeping track of the numerators and denominators separately until the end is optional.

Start with  $t_1 = [\mathbf{P}, 1, 1]$ . Given  $t_j$  and  $t_k$ , to get  $t_{j+k}$ :

form the elliptic curve sum  $j\mathbf{P} + k\mathbf{P} = (j+k)\mathbf{P}$ ;

1 find line( $j\mathbf{P}$ ,  $k\mathbf{P}$ ,  $(-j-k)\mathbf{P}$ )( $\mathbf{X}$ ) =  $c_0 + c_1 * x(\mathbf{X}) + c_2 * y(\mathbf{X})$ ;

2 set:

3 
$$n_{j+k} = n_j * n_k * (c_0 + c_1 * x(\mathbf{X}) + c_2 * y(\mathbf{Q}))$$

4 and

5 
$$d_{j+k} = d_j * d_k * (c_0 + c_1 * x(\mathbf{Q}) - c_2 * y(\mathbf{Q})).$$

6 A similar construction gives  $t_{j-k}$  from  $t_j$  and  $t_k$ . Observe that the vertical  
7 lines through  $(j+k)\mathbf{P}$  and  $(j+k)\mathbf{Q}$  do not appear in the formulae for  $n_{j+k}$  and  $d_{j+k}$ ,  
8 because the contributions from  $\mathbf{Q}$  and  $-\mathbf{Q}$  are equal. When  $j+k=m$ , one can  
9 further simplify this to:

10 
$$n_{j+k} = n_j * n_k \text{ and } d_{j+k} = d_j * d_k,$$

11 since  $c_2$  will be zero.

12 When  $n_m$  and  $d_m$  are nonzero, then the computation

13 
$$\frac{n_m}{d_m} = \frac{f_{m,\mathbf{P}}(\mathbf{Q})}{f_{m,\mathbf{P}}(-\mathbf{Q})}$$

14  
15 has been successful, and after raising to the  $(q-1)/m$  power, one will have the  
16 correct output. If, however, some  $n_m$  or  $d_m$  is zero, then some factor such as  $c_0 +$   
17  $c_1 * x(\mathbf{Q}) + c_2 * y(\mathbf{Q})$  must have vanished. That line was chosen to pass through  $j\mathbf{P}$ ,  
18  $k\mathbf{P}$ , and  $(-j-k)\mathbf{P}$ , for some  $j$  and  $k$ . It does not vanish at any other point on the  
19 elliptic curve. Therefore this factor can vanish only if  $\mathbf{Q} = j\mathbf{P}$  or  $\mathbf{Q} = k\mathbf{P}$  or  $\mathbf{Q} =$   
20  $(-j-k)\mathbf{P}$  for some  $j$  and  $k$ . In all of these cases  $\mathbf{Q}$  will be a multiple of  $\mathbf{P}$ .

21 The above techniques may be implemented through various forms of logic,  
22 including, for example, a programmed computer. Hence, Fig. 4 illustrates a more  
23 general exemplary computer environment 400, which can be used in various  
24 implementations of the invention. The computer environment 400 is only one  
25 example of a computing environment and is not intended to suggest any limitation



1 as to the scope of use or functionality of the computer and network architectures.  
2 Neither should the computer environment 400 be interpreted as having any  
3 dependency or requirement relating to any one or combination of components  
4 illustrated in the exemplary computer environment 400.

5 Computer environment 400 includes a general-purpose computing device in  
6 the form of a computer 402. Computer 402 can implement, for example,  
7 encryptor 102 or decryptor 104 of Fig. 1, generator 120 or client computer 132 of  
8 Fig. 2, either or both of modules 152 and 153 of Fig. 3, and so forth. Computer  
9 402 represents any of a wide variety of computing devices, such as a personal  
10 computer, server computer, hand-held or laptop device, multiprocessor system,  
11 microprocessor-based system, programmable consumer electronics (e.g., digital  
12 video recorders), gaming console, cellular telephone, network PC, minicomputer,  
13 mainframe computer, distributed computing environment that include any of the  
14 above systems or devices, and the like.

15 The components of computer 402 can include, but are not limited to, one or  
16 more processors or processing units 404, a system memory 406, and a system bus  
17 408 that couples various system components including the processor 404 to the  
18 system memory 406. The system bus 408 represents one or more of any of several  
19 types of bus structures, including a memory bus or memory controller, a peripheral  
20 bus, an accelerated graphics port, and a processor or local bus using any of a  
21 variety of bus architectures. By way of example, such architectures can include an  
22 Industry Standard Architecture (ISA) bus, a Micro Channel Architecture (MCA)  
23 bus, an Enhanced ISA (EISA) bus, a Video Electronics Standards Association  
24 (VESA) local bus, and a Peripheral Component Interconnects (PCI) bus also  
25 known as a Mezzanine bus.

1 Computer 402 typically includes a variety of computer readable media.  
2 Such media can be any available media that is accessible by computer 402 and  
3 includes both volatile and non-volatile media, removable and non-removable  
4 media.

5 The system memory 406 includes computer readable media in the form of  
6 volatile memory, such as random access memory (RAM) 410, and/or non-volatile  
7 memory, such as read only memory (ROM) 412. A basic input/output system  
8 (BIOS) 414, containing the basic routines that help to transfer information  
9 between elements within computer 402, such as during start-up, is stored in ROM  
10 412. RAM 410 typically contains data and/or program modules that are  
11 immediately accessible to and/or presently operated on by the processing unit 404.

12 Computer 402 may also include other removable/non-removable,  
13 volatile/non-volatile computer storage media. By way of example, Fig. 4  
14 illustrates a hard disk drive 416 for reading from and writing to a non-removable,  
15 non-volatile magnetic media (not shown), a magnetic disk drive 418 for reading  
16 from and writing to a removable, non-volatile magnetic disk 420 (e.g., a "floppy  
17 disk"), and an optical disk drive 422 for reading from and/or writing to a  
18 removable, non-volatile optical disk 424 such as a CD-ROM, DVD-ROM, or other  
19 optical media. The hard disk drive 416, magnetic disk drive 418, and optical disk  
20 drive 422 are each connected to the system bus 408 by one or more data media  
21 interfaces 425. Alternatively, the hard disk drive 416, magnetic disk drive 418,  
22 and optical disk drive 422 can be connected to the system bus 408 by one or more  
23 interfaces (not shown).

24 The disk drives and their associated computer-readable media provide non-  
25 volatile storage of computer readable instructions, data structures, program

1 modules, and other data for computer 402. Although the example illustrates a hard  
2 disk 416, a removable magnetic disk 420, and a removable optical disk 424, it is to  
3 be appreciated that other types of computer readable media which can store data  
4 that is accessible by a computer, such as magnetic cassettes or other magnetic  
5 storage devices, flash memory cards, CD-ROM, digital versatile disks (DVD) or  
6 other optical storage, random access memories (RAM), read only memories  
7 (ROM), electrically erasable programmable read-only memory (EEPROM), and  
8 the like, can also be utilized to implement the exemplary computing system and  
9 environment.

10 Any number of program modules can be stored on the hard disk 416,  
11 magnetic disk 420, optical disk 424, ROM 412, and/or RAM 410, including by  
12 way of example, an operating system 426, one or more application programs 428,  
13 other program modules 430, and program data 432. Each of such operating  
14 system 426, one or more application programs 428, other program modules 430,  
15 and program data 432 (or some combination thereof) may implement all or part of  
16 the resident components that support the distributed file system.

17 A user can enter commands and information into computer 402 via input  
18 devices such as a keyboard 434 and a pointing device 436 (e.g., a "mouse").  
19 Other input devices 438 (not shown specifically) may include a microphone,  
20 joystick, game pad, satellite dish, serial port, scanner, and/or the like. These and  
21 other input devices are connected to the processing unit 404 via input/output  
22 interfaces 440 that are coupled to the system bus 408, but may be connected by  
23 other interface and bus structures, such as a parallel port, game port, or a universal  
24 serial bus (USB).  
25

1 A monitor 442 or other type of display device can also be connected to the  
2 system bus 408 via an interface, such as a video adapter 444. In addition to the  
3 monitor 442, other output peripheral devices can include components such as  
4 speakers (not shown) and a printer 446 which can be connected to computer 402  
5 via the input/output interfaces 440.

6 Computer 402 can operate in a networked environment using logical  
7 connections to one or more remote computers, such as a remote computing device  
8 448. By way of example, the remote computing device 448 can be a personal  
9 computer, portable computer, a server, a router, a network computer, a peer device  
10 or other common network node, and the like. The remote computing device 448 is  
11 illustrated as a portable computer that can include many or all of the elements and  
12 features described herein relative to computer 402.

13 Logical connections between computer 402 and the remote computer 448  
14 are depicted as a local area network (LAN) 450 and a general wide area network  
15 (WAN) 452. Such networking environments are commonplace in offices,  
16 enterprise-wide computer networks, intranets, and the Internet.

17 When implemented in a LAN networking environment, the computer 402 is  
18 connected to a local network 450 via a network interface or adapter 454. When  
19 implemented in a WAN networking environment, the computer 402 typically  
20 includes a modem 456 or other means for establishing communications over the  
21 wide network 452. The modem 456, which can be internal or external to computer  
22 402, can be connected to the system bus 408 via the input/output interfaces 440 or  
23 other appropriate mechanisms. It is to be appreciated that the illustrated network  
24 connections are exemplary and that other means of establishing communication  
25 link(s) between the computers 402 and 448 can be employed.

1 In a networked environment, such as that illustrated with computing  
2 environment 400, program modules depicted relative to the computer 402, or  
3 portions thereof, may be stored in a remote memory storage device. By way of  
4 example, remote application programs 458 reside on a memory device of remote  
5 computer 448. For purposes of illustration, application programs and other  
6 executable program components such as the operating system are illustrated herein  
7 as discrete blocks, although it is recognized that such programs and components  
8 reside at various times in different storage components of the computing device  
9 402, and are executed by the data processor(s) of the computer.

10 Computer 402 typically includes at least some form of computer readable  
11 media. Computer readable media can be any available media that can be accessed  
12 by computer 402. By way of example, and not limitation, computer readable  
13 media may comprise computer storage media and communication media.  
14 Computer storage media includes volatile and nonvolatile, removable and non-  
15 removable media implemented in any method or technology for storage of  
16 information such as computer readable instructions, data structures, program  
17 modules or other data. Computer storage media include, but are not limited to,  
18 RAM, ROM, EEPROM, flash memory or other memory technology, CD-ROM,  
19 digital versatile disks (DVD) or other optical storage, magnetic cassettes, magnetic  
20 tape, magnetic disk storage or other magnetic storage devices, or any other media  
21 which can be used to store the desired information and which can be accessed by  
22 computer 402. Communication media typically embody computer readable  
23 instructions, data structures, program modules or other data in a modulated data  
24 signal such as a carrier wave or other transport mechanism and includes any  
25 information delivery media. The term "modulated data signal" means a signal that

1 has one or more of its characteristics set or changed in such a manner as to encode  
2 information in the signal. By way of example, and not limitation, communication  
3 media include wired media such as wired network or direct-wired connection, and  
4 wireless media such as acoustic, RF, infrared and other wireless media.  
5 Combinations of any of the above should also be included within the scope of  
6 computer readable media.

7 The invention has been described herein in part in the general context of  
8 computer-executable instructions, such as program modules, executed by one or  
9 more computers or other devices. Generally, program modules include routines,  
10 programs, objects, components, data structures, etc. that perform particular tasks  
11 or implement particular abstract data types. Typically the functionality of the  
12 program modules may be combined or distributed as desired in various  
13 implementations.

14 For purposes of illustration, programs and other executable program  
15 components such as the operating system are illustrated herein as discrete blocks,  
16 although it is recognized that such programs and components reside at various  
17 times in different storage components of the computer, and are executed by the  
18 data processor(s) of the computer.

19 Alternatively, the invention may be implemented in hardware or a  
20 combination of hardware, software, smartcard, and/or firmware. For example, one  
21 or more application specific integrated circuits (ASICs) could be designed or  
22 programmed to carry out the invention.

1 **Conclusion**

2       Although the description above uses language that is specific to structural  
3 features and/or methodological acts, it is to be understood that the invention  
4 defined in the appended claims is not limited to the specific features or acts  
5 described. Rather, the specific features and acts are disclosed as exemplary forms  
6 of implementing the invention.